(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 3 173 884 A1**

(12) EUROPÄISCHE PATENTANMELDUNG

(43) Veröffentlichungstag:
31.05.2017 Patentblatt 2017/22

(51) Int Cl.:
G05B 19/042 *(2006.01)*

(21) Anmeldenummer: **16197755.8**

(22) Anmeldetag: **08.11.2016**

(84) Benannte Vertragsstaaten:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR
Benannte Erstreckungsstaaten:
BA ME
Benannte Validierungsstaaten:
MA MD

(30) Priorität: **24.11.2015 DE 102015120314**

(71) Anmelder: **Pilz GmbH & Co. KG**
73760 Ostfildern (DE)

(72) Erfinder:
- **Wöhrle, Stefan**
  73760 Ostfildern (DE)
- **von Haugwitz, Frank**
  73760 Ostfildern (DE)
- **Bauer, Ralf**
  73760 Ostfildern (DE)

(74) Vertreter: **Witte, Weller & Partner Patentanwälte mbB**
Postfach 10 54 62
70047 Stuttgart (DE)

(54) VERFAHREN ZUM PROGRAMMIEREN EINER SICHERHEITSSTEUERUNG

(57) Zum Programmieren einer Sicherheitssteuerung wird zunächst ein Anwenderprogramm unter Verwendung eines Programmeditors (36) erstellt. Das Anwenderprogramm besitzt einen ersten Programmteil und zumindest einen weiteren Programmteil. Der erste Programmteil definiert sicherheitsrelevante logische Abhängigkeiten zwischen ausgewählten Sensorsignalen und Aktorsignalen. Der erste und der zumindest eine weitere Programmteil werden kompiliert und gebunden, um einen ausführbaren originären Programmcode zu erhalten. Der originäre Programmcode wird in einen ersten Speicher (56) der Sicherheitssteuerung übertragen und mit Hilfe von zumindest einem Prozessor (62a, 62b) ausgeführt. Gemäß einem Aspekt der Erfindung kann der erste Programmteil geändert und kompiliert werden, um einen geänderten ersten Codeteil (61 a) zu erhalten. Der geänderte erste Codeteil (61 a) wird in einen zweiten Speicher (58) der Sicherheitssteuerung übertragen, während der originäre Programmcode mit Hilfe des zumindest einen Prozessors (62a, 62b) ausgeführt wird. Anschließend wird der geänderte erste Codeteil in Ergänzung zu dem originären weiteren Codeteil (48) und anstelle des originären ersten Codeteils (46a) ausgeführt.

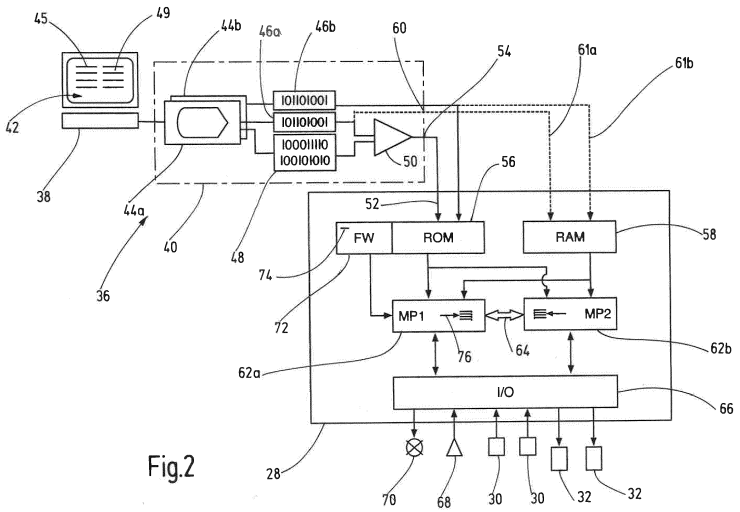

Fig.2

EP 3 173 884 A1

Beschreibung

[0001] Die vorliegende Erfindung betrifft ein Verfahren zum Programmieren einer Sicherheitssteuerung, die eine Vielzahl von Eingängen zum Aufnehmen von Sensorsignalen, eine Vielzahl von Ausgängen zum Ausgeben von Aktorsignalen und zumindest einen Prozessor zum Ausführen von Programmcode aufweist, mit den Schritten:

- Erstellen eines Anwenderprogramms, das logische Abhängigkeiten zwischen den Sensorsignalen und Aktorsignalen definiert, unter Verwendung eines Programmeditors, wobei das Anwenderprogramm einen ersten Programmteil und zumindest einen weiteren Programmteil aufweist, und wobei der erste Programmteil sicherheitsrelevante logische Abhängigkeiten zwischen ausgewählten Sensorsignalen und Aktorsignalen definiert,

- Kompilieren und Binden des ersten und des zumindest einen weiteren Programmteils, um einen ausführbaren originären Programmcode für den zumindest einen Prozessor zu erhalten, wobei der originäre Programmcode einen originären ersten Codeteil aufweist, der den ersten Programmteil repräsentiert, und zumindest einen originären weiteren Codeteil, der den zumindest einen weiteren Programmteil repräsentiert,

- Übertragen des originären Programmcodes in einen ersten Speicher der Sicherheitssteuerung,

- Ausführen des originären Programmcodes mit Hilfe des zumindest einen Prozessors, um die Aktorsignale in Abhängigkeit von den Sensorsignalen zu erzeugen,

- Ändern des ersten Programmteils unter Verwendung des Programmeditors, um einen geänderten ersten Programmteil zu erhalten, und

- Kompilieren des geänderten ersten Programmteils, um einen geänderten ersten Codeteil zu erhalten.

[0002] Die Erfindung betrifft ferner eine Sicherheitssteuerung zum fehlersicheren Steuern eines sicherheitskritischen Prozesses, mit einem Programmiertool, das einen Programmeditor, einen Compiler und eine Schnittstelle zum Übertragen eines originären Programmcodes in einen ersten Speicher der Sicherheitssteuerung entsprechend dem zuvor genannten Verfahren aufweist.

[0003] Eine Sicherheitssteuerung, die sich mit einem solchen Verfahren programmieren lässt, wird von der Pilz GmbH & Co. KG, Felix-Wankel-Straße 2, 72760 Ostfildern, Deutschland unter der Bezeichnung Automatisierungssystem PSS 4000 angeboten und ist beispielsweise in einer Broschüre mit dem Titel "Automation System PSS 4000 - Building Block System" aus dem Jahr 2015 beschrieben. Des Weiteren offenbart DE 10 2009 019 096 A1 eine solche Sicherheitssteuerung.

[0004] Die bekannte Sicherheitssteuerung besteht aus einer Vielzahl von Hardware- und Softwarekomponenten, aus denen ein Anwender ein Automatisierungssystem nach seinen individuellen Bedürfnissen zusammenstellen kann. Je nach Größe der Anwendung kann das System eine Vielzahl an Steuereinheiten beinhalten, die über ein Kommunikationsnetzwerk miteinander verbunden sind und jeweils Teilprozesse innerhalb einer komplexen Anlage steuern. Die vorliegende Erfindung ist allerdings nicht auf ein komplexes System mit einer Vielzahl von vernetzten Steuereinheiten beschränkt und kann gleichermaßen zum Programmieren einer "kleinen" Sicherheitssteuerung verwendet werden, die in kompakter Bauform einige wenige Sicherheitsfunktionen in einer automatisiert betriebenen Anlage oder Maschine steuert.

[0005] Des Weiteren ist die vorliegende Erfindung nicht auf eine "reine" Sicherheitssteuerung beschränkt. Sie kann vielmehr auch bei Steuerungen verwendet werden, die sicherheitsrelevante Prozesse (sogenannte FS- bzw. Failsafe-Prozesse) und nicht-sicherheitsrelevante Prozesse (sogenannte Standard-Prozesse) gleichermaßen steuern. Ein nichtsicherheitsrelevanter Standard-Prozess betrifft typischerweise den normalen Betriebsablauf einer automatisierten Anlage. Fehler in der Steuerung dieses Prozesses mögen aus wirtschaftlichen Gründen ärgerlich sein und sollen daher möglichst vermieden werden. Sie gefährden jedoch nicht die Gesundheit oder das Leben von Bedienpersonen oder anderen Personen in der Umgebung der Anlage. Im Unterschied dazu dienen sicherheitsrelevante Steuerfunktionen primär dazu, die Gefahren, die von einer automatisiert betriebenen Anlage ausgehen, zu beherrschen. Aus diesem Grund werden an die Steuerung von sicherheitsrelevanten Prozessen in der Automatisierungstechnik besondere Anforderungen im Hinblick auf die Vermeidung und Beherrschung von Fehlern gestellt. Sicherheitsrelevante Steuerungen und Steuerkomponenten erfordern einen höheren Entwicklungsaufwand und benötigen typischerweise besondere Zulassungen, insbesondere nach den Normen EN ISO 13849, EN/IEC 61508, EN IEC 62061 u.a.

[0006] Das Automatisierungssystem PSS 4000 ist einerseits eine Sicherheitssteuerung im Sinne der vorgenannten Normen, da es zahlreiche Komponenten beinhaltet, die die erforderlichen Zulassungen zum Steuern von sicherheitskritischen Prozessen besitzen. Insbesondere erfüllen zahlreiche Komponenten die Anforderungen gemäß SIL 2 und höher nach EN/IEC 61508 und/oder PL d gemäß EN ISO 13849 und höher. Darüber hinaus ist das Automatisierungssystem PSS 4000 jedoch auch dazu ausgelegt, Standard-Prozesse einer komplexen Anlage zu steuern. Die Programmierung der bekannten Sicherheitssteuerung kann daher im Einzelfall sehr aufwendig und umfangreich sein, wenn nämlich innerhalb einer komplexen Anlage eine große Zahl an

sicherheitsrelevanten und nichtsicherheitsrelevanten Prozessen gesteuert werden muss.

[0007] Es ist daher typisch für solche Systeme, dass das sogenannte Anwenderprogramm, das die logischen Abhängigkeiten zwischen den Sensorsignalen und Aktorsignalen innerhalb der Anlage definiert, in mehreren Teilen oder Modulen erstellt wird, wobei der Anwender in der Regel eine höhere Programmiersprache verwendet, die auf der internationalen Norm IEC 61131 basiert. Die einzelnen Programmteile werden anschließend zusammen kompiliert, d.h. in einen maschinenlesbaren Code übersetzt, und gebunden. Der erhaltene Programmcode wird dann in alle Steuereinheiten heruntergeladen, die den entsprechenden Programmcode ausführen sollen.

[0008] Die Erstellen eines komplexen Anwenderprogramms ist häufig nicht in einem einzigen Schritt möglich. In der Regel ergibt sich nach Inbetriebnahme der Sicherheitssteuerung ein Änderungs- und/oder Ergänzungsbedarf, weil man erst bei laufender Anlage alle Zusammenhänge erkennen kann. Bei der bekannten Sicherheitssteuerung kann der Anwender einen oder mehrere Teile des Anwenderprogramms in der höheren Programmiersprache ändern. Anschließend muss das geänderte Anwenderprogramm erneut kompiliert werden und die kompilierten Teile müssen erneut gebunden werden, um einen maschinenlesbaren Programmcode mit den entsprechenden Änderungen zu erhalten. Der geänderte Programmcode wird dann erneut in die Sicherheitssteuerung geladen und überschreibt den vorherigen (originären) Programmcode. Die Durchführung einer Änderung erfordert daher bei der bekannten Sicherheitssteuerung einen Stopp des Steuerungsablaufs, um das Herunterladen des geänderten Programmcodes zu ermöglichen. Anschließend muss die Anlage neu gestartet werden, was insbesondere bei komplexen Anlagen zeitaufwendig sein kann.

[0009] Bei einer früheren Steuerung der Pilz GmbH & Co. KG, die unter der Bezeichnung PSS 3000 angeboten wurde, war es möglich, Standard-Bausteine auch bei laufender Maschine in den Speicher der Steuerung zu übertragen. Dies erleichtert eine nachträgliche Änderung an einem Anwenderprogramm. Allerdings beschränkte sich diese Updatemöglichkeit auf Programmteile, die ausschließlich nicht-sicherheitsrelevante Standard-Prozesse betrafen. Der praktische Nutzen dieser Funktion war daher begrenzt.

[0010] Vor diesem Hintergrund ist es eine Aufgabe der vorliegenden Erfindung, ein Verfahren der eingangs genannten Art anzugeben, das eine schnellere Änderung des Programmcodes für die Sicherheitssteuerung ermöglicht. Es ist eine weitere Aufgabe der Erfindung, eine Sicherheitssteuerung anzugeben, die sich mit einem solchen Verfahren einfacher und schneller programmieren lässt.

[0011] Gemäß einem ersten Aspekt der Erfindung wird diese Aufgabe durch ein Verfahren der eingangs genannten Art gelöst, wobei der geänderte erste Codeteil in einen zweiten Speicher der Sicherheitssteuerung übertragen wird, während der originäre Programmcode mit Hilfe des zumindest einen Prozessors ausgeführt wird, und wobei der geänderte erste Codeteil anschließend in Ergänzung zu dem originären weiteren Codeteil und anstelle des originären ersten Codeteils mit Hilfe des zumindest einen Prozessors ausgeführt wird, um die Aktorsignale in Abhängigkeit von dem geänderten ersten Programmteil zu erzeugen.

[0012] Gemäß einem weiteren Aspekt wird diese Aufgabe durch eine Sicherheitssteuerung der eingangs genannten Art gelöst, wobei das Programmiertool dazu ausgebildet ist, den ersten Programmteil separat von dem weiteren Programmteil zu ändern und zu kompilieren, um einen geänderten ersten Codeteil zu erhalten, und wobei das Programmiertool eine Schnittstelle aufweist, die dazu ausgebildet ist, den geänderten ersten Codeteil in den zweiten Speicher zu übertragen, während der originäre Programmcode mit Hilfe des zumindest einen Prozessors ausgeführt wird, und wobei die Sicherheitssteuerung dazu ausgebildet ist, den geänderten ersten Codeteil in Ergänzung zu dem originären weiteren Codeteil und anstelle des originären ersten Codeteils mit Hilfe des zumindest einen Prozessors auszuführen, um die Aktorsignale in Abhängigkeit von dem geänderten ersten Programmteil zu erzeugen.

[0013] Besonders vorteilhaft lassen sich das neue Verfahren und die neue Sicherheitssteuerung mit einem Computerprogramm realisieren, das den vorgenannten Verfahrensablauf ermöglicht, sobald das Computerprogramm auf der Sicherheitssteuerung implementiert ist.

[0014] Das neue Verfahren und die entsprechende Sicherheitssteuerung ermöglichen eine Änderung des ersten Programmteils unabhängig von Änderungen in dem zumindest einen weiteren Programmteil sowie ein anschließendes isoliertes Kompilieren des geänderten ersten Programmteils, um auf diese Weise einen geänderten ersten Codeteil zu erzeugen, der als isoliertes Programmcodefragment in den zweiten Speicher der Sicherheitssteuerung geladen wird. Vorzugsweise wird der geänderte erste Codeteil ohne vorheriges Binden mit anderen Codeteilen in den zweiten Speicher geladen. Da der erste Speicher mit dem originären Programmcode hierdurch nicht überschrieben wird, ist das Herunterladen des geänderten ersten Codeteils ohne Weiteres möglich, während der originäre Programmcode mit Hilfe des zumindest einen Prozessors ausgeführt wird. Das Herunterladen des geänderten ersten Codeteils erfordert daher keine Unterbrechung bzw. keinen Stopp des Steuerungsablaufs. Anders ausgedrückt kann der erste Programmteil geändert werden und kann der geänderte erste Codeteil in den zweiten Speicher der Sicherheitssteuerung übertragen werden, während die Sicherheitssteuerung den originären Programmcode ausführt und folglich den - zumindest teilweise - sicherheitskritischen Prozess in der automatisierten Anlage steuert. Das isolierte Kompilieren des geänderten ersten Programmteils und der bevorzugte Verzicht auf das Binden mit dem zumin-

dest einen weiteren Programmteil ermöglicht die schnellere Erzeugung eines maschinenlesbaren geänderten Codeteils. Allein dadurch ist bereits eine schnellere Änderung in der Programmierung der Sicherheitssteuerung möglich.

[0015] Darüber hinaus kann der geänderte erste Codeteil von der Sicherheitssteuerung jedoch ausgeführt werden, ohne den Steuerungsablauf zu unterbrechen. Vorteilhaft kann die Sicherheitssteuerung den geänderten ersten Codeteil unter Verwendung der aktuellen Prozesswerte der gesteuerten Anlage ausführen. Dies ermöglicht eine große Zeitersparnis, da die Sicherheitssteuerung und die gesteuerte Anlage vor einem Neustart zunächst in einen definierten Zustand gebracht werden müssen, was bei einer komplexen Anlage sehr zeitaufwendig sein kann. Dies kann aufgrund des neuen Verfahrens entfallen.

[0016] Die Übernahme des geänderten ersten Codeteils im laufenden Betrieb der Sicherheitssteuerung ist in einigen vorteilhaften Ausführungsbeispielen dadurch realisiert, dass die Sicherheitssteuerung von dem Programmiertool veranlasst wird, einen Programmzeiger so zu ändern, dass der geänderte erste Codeteil aus dem zweiten Speicher anstelle des originären ersten Codeteils aus dem ersten Speicher ausgeführt wird, sobald innerhalb der zyklischen Ausführung des Programmcodes die Ausführung des ersten Codeteils ansteht. In einigen vorteilhaften Ausführungsbeispielen erfolgt die Änderung des Programmzeigers am Ende eines Programmzyklus, so dass der geänderte erste Codeteil von einem Programmzyklus zum nächsten ausgeführt wird. Die aktuellen Prozesswerte aller Variablen bleiben dabei vorteilhaft erhalten. Prinzipiell wäre es jedoch denkbar, dass der originäre erste Codeteil mit dem geänderten ersten Codeteil überschrieben wird, insbesondere wenn der Programmcode nicht direkt aus dem ersten Speicher ausgeführt wird, sondern beispielsweise aus einem speziellen Arbeitsspeicher.

[0017] In allen bevorzugten Ausführungsbeispielen bleiben sämtliche I/O-Zuordnungen zwischen den physischen Eingängen und Ausgängen der Sicherheitssteuerung und den im Programmcode verwendeten Variablen erhalten. Bevorzugt beschränken sich die Änderungen in dem geänderten ersten Codeteil daher allein auf geänderte logische Abhängigkeiten zwischen den ausgewählten Sensorsignalen und Aktorsignalen. Dies erleichtert die schnelle Übernahme des geänderten Codeteils in den Steuerungsablauf.

[0018] Da der geänderte erste Codeteil sicherheitsrelevante logische Abhängigkeiten zwischen ausgewählten Sensorsignalen und Aktorsignalen repräsentiert, ermöglichen das neue Verfahren und die neue Sicherheitssteuerung eine Änderung des sicherheitsrelevanten Steuerungsablaufs im laufenden Betrieb der Sicherheitssteuerung. Dies ist einerseits sehr vorteilhaft, weil sicherheitsrelevante Steuerungsanteile die Steuerung des nicht-sicherheitsrelevanten Betriebsablaufs der Anlage erschweren und beeinträchtigen können. Infolgedessen ermöglichen das neue Verfahren und die entsprechende Sicherheitssteuerung beispielsweise eine zeitlich begrenzte Außerkraftsetzung eines sicherheitsrelevanten Steuerungsablaufs, um etwa eine Fehlersuche im nicht-sicherheitsrelevanten Steuerungsablauf zu erleichtern.

[0019] In vorteilhaften Ausführungsbeispielen ermöglichen das neue Verfahren und die neue Sicherheitssteuerung darüber hinaus auch eine Änderung in dem zumindest einen weiteren Programmteil während des laufenden Betriebs, indem der zumindest eine weitere Programmteil separat geändert und kompiliert wird und der geänderte weitere Codeteil in den zweiten Speicher übertragen wird. Die Besonderheit des neuen Verfahrens und der neuen Sicherheitssteuerung liegt ungeachtet dessen jedoch darin, dass der erste Programmteil inklusive der sicherheitsrelevanten logischen Abhängigkeiten in der beschriebenen Weise isoliert geändert, kompiliert und in den zweiten Speicher der Sicherheitssteuerung übertragen werden kann. In einigen vorteilhaften Ausführungsbeispielen ist die Änderung des ersten Programmteils und die Übertragung des geänderten ersten Codeteils in den zweiten Speicher der Sicherheitssteuerung daher nur nach einer erfolgreichen Identifizierung und/oder Autorisierung des Anwenders und einer entsprechenden Freigabe durch das Betriebssystem der Sicherheitssteuerung möglich.

[0020] Des Weiteren ist es in einigen vorteilhaften Ausführungsbeispielen möglich, die Ausführung des geänderten ersten Codeteils aus dem zweiten Speicher während der Ausführung des originären Programmcodes zu sperren, wobei diese Sperre vorteilhaft bei der Ausführung des originären Programmcodes zweikanalig überwacht wird. Diese Ausführungsbeispiele verhindern somit jeglichen Eingriff in den sicherheitsrelevanten Steuerungsablauf in Abhängigkeit von dem aktuell ausgeführten originären Programmcode, was eine erhöhte Sicherheit bietet, aber die Durchführung von Änderungen im Einzelfall erschwert. Vorteilhaft besitzen das neue Verfahren und die neue Sicherheitssteuerung grundsätzlich die Möglichkeit, einen isoliert kompilierten ersten Codeteil in den zweiten Speicher der Sicherheitssteuerung zu laden und den zumindest einen Prozessor zur Ausführung dieses Codeteils im laufenden Betrieb zu veranlassen, wobei diese grundsätzlich bestehende Möglichkeit deaktiviert werden kann.

[0021] Insgesamt ermöglichen das neue Verfahren und die neue Sicherheitssteuerung eine schnellere Übernahme eines geänderten Codeteils, der sicherheitsrelevante logische Abhängigkeiten repräsentiert. Damit kann ein Anwender die neue Sicherheitssteuerung schneller umprogrammieren als dies bisher der Fall gewesen ist. Die oben genannte Aufgabe ist daher vollständig gelöst.

[0022] In einer bevorzugten Ausgestaltung der Erfindung wird der geänderte erste Codeteil nur während einer begrenzten Zeitspanne anstelle des originären ersten Codeteils ausgeführt.

[0023] In dieser Ausgestaltung ist eine Änderung des Programmcodes nach dem neuen Verfahren nur tempo-

rär möglich. Nach Ablauf der begrenzten Zeitspanne führt der zumindest eine Prozessor daher wieder den originären ersten Codeteil anstelle des geänderten ersten Codeteils aus. Dies kann in einigen Ausführungsbeispielen automatisch erfolgen. Bevorzugt erfordert die Rückkehr zu dem originären ersten Codeteil jedoch einen manuellen Eingriff, wie weiter unten anhand eines bevorzugten Ausführungsbeispiels erläutert ist. In einigen vorteilhaften Ausführungsbeispielen ist die begrenzte Zeitspanne ≤ 24 Stunden, vorteilhaft ≤ 12 Stunden. In weiteren vorteilhaften Ausführungsbeispielen kann die begrenzte Zeitspanne ≤ 5 Stunden oder gar ≤ 0,5 Stunden sein. Die Ausgestaltung besitzt den Vorteil, dass durch das neue Verfahren keine dauerhafte Änderung des sicherheitsrelevanten Programmcodes vorgenommen werden kann, ohne die etablierte und bewährte Vorgehensweise dafür zu umgehen. Infolgedessen ist die Fehlersicherheit im sicherheitsrelevanten Steuerungsablauf erhöht. Insbesondere ist eine dauerhafte Herabsetzung des Sicherheitsniveaus, die bei der Inbetriebnahme einer Anlage zur erleichterten Fehlersuche von Vorteil sein kann, zeitlich begrenzt. Auf Dauer arbeitet die neue Sicherheitssteuerung mit dem originären Programmcode.

[0024] In einer weiteren Ausgestaltung nimmt die Sicherheitssteuerung nach Ablauf der begrenzten Zeitspanne einen definierten sicheren Zustand ein.

[0025] Ein definierter sicherer Zustand im Sinne dieser Ausgestaltung ist ein Zustand, in dem gefährliche Aktoren in ihren sicheren Zustand gebracht sind. In der Regel erfordert der sichere Zustand einen manuellen Reset oder einen manuellen Neustart der Sicherheitssteuerung für die erneute Inbetriebnahme. In bevorzugten Ausführungsbeispielen ist die Überwachung der begrenzten Zeitspanne in der Firmware der Sicherheitssteuerung implementiert, bevorzugt zweikanalig. Vorteilhaft kann sie vom Anwender nicht außer Kraft gesetzt werden. Die Ausgestaltung bietet eine weiter erhöhte Sicherheit in Bezug auf die sicherheitsrelevante Steuerung.

[0026] In einer weiteren Ausgestaltung ist der erste Speicher ein nichtflüchtiger Speicher und der zweite Speicher ein flüchtiger Speicher. Ein nichtflüchtiger Speicher im Sinne dieser Ausgestaltung ist ein Speicher, der den übertragenen Programmcode auch bei Wegfall der Betriebs- oder Versorgungsspannung behält, während ein flüchtiger Speicher die eingespeicherten Codeteile bei Wegfall der Betriebs- oder Versorgungsspannung verliert. Die Ausgestaltung besitzt den Vorteil, dass ein Neustart - und vorzugsweise auch ein manueller Reset - die Nutzung des geänderten ersten Codeteils zuverlässig beendet. Die durch das neue Verfahren gewonnene Flexibilität und Erleichterung bei der Fehlersuche ist durch den flüchtigen Charakter des zweiten Speichers zugunsten einer besseren Absicherung der Anlage begrenzt.

[0027] In einer weiteren Ausgestaltung wird ein Programmzeiger der Sicherheitssteuerung geändert, um den geänderten ersten Codeteil anstelle des originären ersten Codeteils auszuführen.

[0028] In dieser Ausgestaltung besitzt die Sicherheitssteuerung beide erste Codeteile, gewissermaßen in Koexistenz. Durch Änderung des Programmzeigers, der in an sich bekannter Weise auf den jeweils nächstfolgenden Maschinencodebefehl im Speicher der Steuerung zeigt, ist ein sehr schnelles Umschalten auf den geänderten ersten Codeteil möglich. Des Weiteren kann aufgrund der Koexistenz der beiden Codeteile ohne Weiteres auf den originären Codeteil zurückgewechselt werden. Alternativ hierzu wäre es grundsätzlich aber denkbar, dass der geänderte erste Codeteil aus dem zweiten Speicher in den ersten Speicher übertragen wird und dort den originären ersten Codeteil überschreibt. Die bevorzugte Ausgestaltung vereinfacht demgegenüber die oben erwähnte, temporäre Ausführung des geänderten ersten Codeteils.

[0029] In einer weiteren Ausgestaltung führt der zumindest eine Prozessor den originären Programmcode zyklisch aus, wobei der geänderte erste Codeteil nach Abschluss eines Programmzyklus anstelle des originären ersten Codeteils ausgeführt wird.

[0030] Diese Ausgestaltung ermöglicht einen definierten nahtlosen Übergang von dem originären ersten Codeteil zu dem geänderten ersten Codeteil im laufenden Betrieb der Sicherheitssteuerung. Vorteilhaft erhält der geänderte erste Codeteil in dieser Ausgestaltung aktuelle Prozesswerte, so dass der Steuerungsablauf trotz der Änderung störungsfrei ausgeführt werden kann.

[0031] In einer weiteren Ausgestaltung weist die Sicherheitssteuerung eine optische Anzeige auf, mit der die Ausführung des geänderten ersten Codeteils signalisiert wird. In vorteilhaften Ausführungsbeispielen signalisiert die optische Anzeige die Ausführung des geänderten ersten Codeteils während der gesamten Zeitspanne beginnend ab der Übernahme des geänderten ersten Codeteils bis zum Ablauf der begrenzten Zeitspanne und/oder bis zu einem manuellen Reset oder Neustart, mithin also solange der geänderte erste Codeteil "aktiv" ist. Die Ausgestaltung ist im Hinblick auf die Fehlersicherheit des gesteuerten Prozessablaufs vorteilhaft, da die Abweichung vom originären Programmcode eine Änderung des Sicherheitsniveaus zur Folge haben kann und in dieser Ausgestaltung vorteilhaft kenntlich gemacht ist.

[0032] In einer weiteren Ausgestaltung wird der geänderte erste Programmteil zusammen mit dem zumindest einen weiteren Programmteil kompiliert und gebunden, um einen geänderten ausführbaren Programmcode zu erhalten, und der geänderte ausführbare Programmcode wird in den ersten Speicher übertragen, nachdem die Ausführung des originären Programmcodes gestoppt wurde.

[0033] In dieser Ausgestaltung wird der geänderte erste Programmteil dauerhaft in den ausführbaren Programmcode eingebunden und gewissermaßen als "neuer originärer Programmcode" ohne zeitliche Begrenzung in die Sicherheitssteuerung geladen. Diese Ausgestal-

tung ist vorteilhaft, um eine Änderung im ersten Programmteil nach einer ersten Testphase dauerhaft in der neuen Sicherheitssteuerung zu implementieren. Vorteilhaft überschreibt der geänderte ausführbare Programmcode den originären ausführbaren Programmcode in dem ersten Speicher.

[0034] In einer weiteren Ausgestaltung wird der geänderte erste Programmteil zweifach kompiliert, um einen geänderten ersten Codeteil und einen geänderten zweiten Codeteil zu erzeugen, wobei der geänderte erste Codeteil und der geänderte zweite Codeteil in den zweiten Speicher übertragen werden und mit Hilfe des zumindest einen Prozessors redundant ausgeführt werden.

[0035] Diese Ausgestaltung ist besonders vorteilhaft, um eine Einfehlersicherheit bei der Ausführung des sicherheitsrelevanten ersten Programmteils auch nach der Änderung zu gewährleisten. Vorteilhaft wird auch der originäre erste Programmteil zweifach kompiliert, wobei der geänderte erste Codeteil und der geänderte zweite Codeteil in der beschriebenen Weise anstelle des originären ersten und eines originären zweiten Codeteils ausgeführt werden. Bevorzugt ist ferner, wenn der erste Codeteil und der zweite Codeteil auf separaten Prozessoren ausgeführt werden, um auf diese Weise zwei voneinander getrennte Kanäle zu realisieren, die im Sinne einer mehrkanaligen, redundanten Ausführung arbeiten. Die Ausgestaltung bietet eine hohe Fehlersicherheit, die gerade auch über die neu geschaffene Möglichkeit zur schnellen Änderung eines sicherheitsrelevanten Programmteils erhalten bleibt. Darüber hinaus besitzt diese Ausgestaltung den Vorteil, dass der geänderte erste Programmteil unter realen Bedingungen in der Sicherheitssteuerung ausgeführt wird, was eine dauerhafte Übernahme im Sinne der oben beschriebenen Ausgestaltung erleichtert.

[0036] In einer weiteren Ausgestaltung ist die Sicherheitssteuerung eine verteilte Sicherheitssteuerung mit einer Vielzahl von vernetzten Steuereinheiten, auf denen jeweils individueller Programmcode ausgeführt wird, wobei der geänderte erste Codeteil an zumindest zwei verschiedene Steuereinheiten übertragen wird, während der originäre Programmcode ausgeführt wird. In bevorzugten Ausführungsbeispielen beinhaltet die verteilte Sicherheitssteuerung einen zentralen Speicher, in dem der geänderte erste Codeteil bereitgestellt wird, und die zumindest zwei verschiedenen Steuereinheiten laden den geänderten ersten Codeteil aus dem zentralen Speicher in ihren jeweiligen ersten Speicher. Diese Ausgestaltung ermöglicht auf einfache, komfortable und schnelle Weise die Änderung eines komplexen aspekt- und/oder objektorientierten Anwenderprogramms in einer verteilten Sicherheitssteuerung, wobei die Änderungen unter realen Bedingungen in mehreren Steuereinheiten ausgeführt und somit getestet werden können. In vorteilhaften Ausführungsbeispielen ist das Programmiertool dazu ausgebildet, alle vernetzten Steuereinheiten zu ermitteln, auf denen der originäre erste Programmteil ausgeführt wird, um anschließend alle betroffenen Steuereinheiten zu veranlassen, den geänderten ersten Codeteil in den jeweiligen zweiten Speicher zu laden und in der beschriebenen Weise auszuführen. In einigen bevorzugten Ausführungsbeispielen ist das Programmiertool dazu ausgebildet, den geänderten ersten Programmteil für die zumindest zwei verschiedenen Steuereinheiten unterschiedlich zu kompilieren, um den geänderten ersten Programmteil auf unterschiedliche Hardwareplattformen auszuführen.

[0037] In einer weiteren Ausgestaltung wird die Änderung des ersten Programmteils in Abhängigkeit von einer Anwenderidentifikation freigegeben oder gesperrt.

[0038] In dieser Ausgestaltung muss ein Anwender, der den ersten Programmteil ändern möchte, zunächst eine Identifikationsprozedur erfolgreich durchlaufen. In einigen Ausführungsbeispielen kann die Anwenderidentifizierung die Eingabe eines Passworts beinhalten. Alternativ oder ergänzend kann die Anwenderidentifizierung anhand eines elektronischen und/oder mechanischen Schlüssels erfolgen. Die Ausgestaltung bietet eine erhöhte Sicherheit gegen Manipulationen und unbewusste Eingriffe in den sicherheitsrelevanten Steuerungsablauf.

[0039] In einer weiteren Ausgestaltung wird die Übertragung des geänderten ersten Codeteils an den zweiten Speicher in einem weiteren Speicher mit einem Datumsstempel protokolliert. Vorzugsweise ist der weitere Speicher ein nichtflüchtiger Speicher.

[0040] Die Ausgestaltung besitzt den Vorteil, dass jede Veränderung an dem sicherheitsrelevanten ersten Programmteil automatisch dokumentiert wird, auch wenn sie nur temporär möglich ist, und somit auch später nachvollzogen werden kann. Dies erleichtert zum einen die Fehlersuche und Optimierung des Programmcodes bei der Programmierung der Sicherheitssteuerung. Andererseits lassen sich Manipulationen oder Manipulationsversuche leichter erkennen und unterbinden.

[0041] Es versteht sich, dass die vorstehend genannten und die nachstehend noch zu erläuternden Merkmale nicht nur in der jeweils angegebenen Kombination, sondern auch in anderen Kombinationen oder in Alleinstellung verwendbar sind, ohne den Rahmen der vorliegenden Erfindung zu verlassen.

[0042] Ausführungsbeispiele der Erfindung sind in der Zeichnung dargestellt und werden in der nachfolgenden Beschreibung näher erläutert. Es zeigen:

Fig. 1 eine sehr schematische Darstellung einer komplexen Anlage mit einer verteilten Sicherheitssteuerung nach einem Ausführungsbeispiel der Erfindung;

Fig. 2 eine schematische Darstellung einer einzelnen Steuereinheit der Anlage aus Fig. 1; und

Fig. 3 ein Flussdiagramm zur Erläuterung eines Ausführungsbeispiels des neuen Verfahrens.

[0043] In Fig. 1 ist eine Anlage, die mit einem Ausfüh-

rungsbeispiel der neuen Sicherheitssteuerung gesteuert wird, in ihrer Gesamtheit mit der Bezugsziffer 10 bezeichnet. Die Anlage 10 beinhaltet eine Vielzahl von Anlagenhardwarekomponenten 12. Im vorliegenden Ausführungsbeispiel sind dies eine Bestückungsstation 14, eine Bearbeitungsstation 16, eine Teststation 18, eine Fördereinheit 20 und eine Verpackungs- und Palettierstation 22. Mit der Bestückungsstation 14, die beispielsweise einen Roboter beinhalten kann, werden der Bearbeitungsstation 16 Werkstücke automatisiert zugeführt. Diese Werkstücke werden in der Bearbeitungsstation 16 bearbeitet. Anschließend werden die bearbeiteten Werkstücke an die Teststation 18 weitergegeben, in der überprüft wird, ob das bearbeitete Werkstück vordefinierte Prüfkriterien erfüllt. Sind diese Prüfkriterien erfüllt, wird das Werkstück mittels der Fördereinheit 20 an die Verpackungs- und Palettierstation 22 übergeben. In dieser werden mehrere Werkstücke zu Gebinden zusammengefasst, die dann auf einer Palette gestapelt werden. Die Anlage 10 steht somit exemplarisch für eine Vielzahl von Anlagen, die automatisiert gesteuert werden, wobei sicherheitsrelevante und nicht-sicherheitsrelevante Prozesse zu einem Gesamtablauf kombiniert sind,

[0044] Mit der Bezugsziffer 24 ist hier eine Sicherheitssteuerung gemäß einem Ausführungsbeispiel der Erfindung in ihrer Gesamtheit bezeichnet. Die Sicherheitssteuerung 24 beinhaltet eine Vielzahl von Hardwarekomponenten 26, insbesondere in Form von programmierbaren Steuereinheiten 28, Sensorgeräten 30 und Aktorgeräten 32, sowie mehrere Softwarekomponenten einschließlich eines Anwenderprogramms, das hier nach dem neuen Verfahren erstellt wird. Die Sensorgeräte 30 und Aktorgeräte 32 sind in bevorzugten Ausführungsbeispielen modular zusammengestellte I/O-Geräte, an die eine Vielzahl verschiedener Sensoren und Aktoren angeschlossen werden können, wie etwa Positionsschalter, Drehgeber, Temperatursensoren, Magnetventile, Schütze und/oder elektrische Antriebe. In bevorzugten Ausführungsbeispielen kann eine Steuereinheit zusammen mit einem modularen Sensor- und Aktorgerät eine kombinierte Baugruppe bilden, wie dies prinzipiell von dem eingangs genannten Automatisierungssystem PSS 4000 bekannt ist.

[0045] Vorteilhaft sind die Steuereinheiten, Sensoren und Aktoren jeweils einer der Anlagenhardwarekomponenten 12 zugeordnet. Die Steuerungshardwarekomponenten 26 sind vorteilhaft über ein Kommunikationsnetzwerk 34 miteinander verbunden. Das Kommunikationsnetzwerk kann ein Ethernet-basiertes Bussystem beinhalten, wie es von der Pilz GmbH & Co. KG unter der Bezeichnung SafetyNET p® angeboten und vertrieben wird. Alternativ oder ergänzend kann das Kommunikationsnetzwerk 34 weitere Bussysteme beinhalten, die insbesondere für den zyklischen Datenverkehr in einem automatisierten Steuerungssystem ausgebildet sind.

[0046] Die Arbeitsbereiche der einzelnen Stationen 14, 16, 18, 22 können beispielsweise durch Schutztüren gesichert sein, die einen Zugang in den gefährlichen Arbeitsbereich der jeweiligen Station nur zulassen, wenn die zugeordnete Steuereinheit die Station in einen sicheren Zustand gesteuert hat. Alternativ oder ergänzend können Lichtgitter oder Lichtvorhänge eingesetzt werden. Darüber hinaus können die einzelnen Stationen 14, 16, 18, 22 mit Not-Aus-Tastern versehen sein, mit denen die jeweilige Station in einen sicheren Zustand überführt werden kann, insbesondere durch Trennen von der Stromversorgung. Hierzu werden in der Regel Schütze angesteuert, die im Stromversorgungspfad zu der jeweiligen Station angeordnet sind.

[0047] Bei den vorstehend genannten Schutztüren, Lichtgittern, Lichtvorhängen und Not-Aus-Tastern handelt es sich um typische sicherheitsrelevante Sensoren, deren Ausgangssignale logisch miteinander verknüpft werden, um in Abhängigkeit davon sicherheitsrelevante Aktoren, wie etwa die Schütze im Stromversorgungspfad der Stationen anzusteuern. Die Sensoren 30 einer Station können darüber hinaus zahlreiche nicht-sicherheitsrelevante Sensoren beinhalten, die für die Steuerung des Betriebsablaufs benötigt werden und beispielsweise betriebsmäßige Drehzahlen, Winkel, Positionen oder Geschwindigkeiten erfassen, die für die Steuerung des Betriebsablaufs benötigt werden. Die Aktoren 32 können gleichermaßen nicht-sicherheitsrelevante Aktoren beinhalten, insbesondere Motoren oder Stellzylinder.

[0048] Im vorliegenden Ausführungsbeispiel ist jeder Station 14, 16, 18, 22 eine eigene Steuereinheit 28 zugeordnet. Es ist alternativ möglich, mehreren Stationen einer Anlage mit einer gemeinsamen Steuereinheit zu steuern. Die Anlagenkomponenten können baulich und räumlich voneinander getrennt sein. Es ist aber auch denkbar, dass mehrere Anlagenkomponenten zu einer integrierten Station verbunden sind.

[0049] In Fig. 1 sind funktionsgleiche Komponenten mit gleichen Bezugsziffern bezeichnet, wobei Striche andeuten ist, dass die jeweiligen Komponenten unterschiedlich realisiert sein können. Entsprechendes gilt auch für Signale. Diese Form der Kennzeichnung gilt auch für die weiteren Figuren.

[0050] In Fig. 2 ist eine einzelne Steuereinheit 28 zusammen mit einem Programmiertool 36 dargestellt. Das Programmiertool ist in bevorzugten Ausführungsbeispielen ein PC 38, auf dem ein Computerprogramm (Software) 40 ausgeführt wird. Das Computerprogramm 40 stellt auf dem PC 38 einen Programmeditor 42 bereit, der es einem Anwender ermöglicht, ein Anwenderprogramm für die Steuereinheit 28 und/oder für eine Vielzahl von Steuereinheiten 28, 28', 28" etc. in einer für den Anwender komfortablen Programmiersprache zu erstellen. Vorzugsweise stellt der Programmeditor eine oder mehrere Programmiersprachen zur Verfügung, die auf der Norm EN 61131 basieren. Hierzu gehört beispielsweise eine Programmiersprache, die die Definition von logischen Abhängigkeiten zwischen ausgewählten Sensorsignalen und ausgewählten Aktorsignalen in grafischer Form ermöglicht, insbesondere anhand von sogenannten Funktionsblöcken. Das Programmiertool 36 beinhaltet

ferner einen Compiler 44, mit dessen Hilfe ein in einer höheren Programmiersprache erstellter Programmteil in einen maschinenlesbaren und von der Steuerung 28 ausführbaren Maschinencode übersetzt werden kann. In bevorzugten Ausführungsbeispielen beinhaltet das Programmiertool einen ersten Compiler 44a und einen zweiten Compiler 44b, die aus einem sicherheitsrelevanten Programmteil 45 zwei separate ausführbare Programmcodes erzeugen, die anschließend von einem oder mehreren Prozessoren redundant ausgeführt werden, um eine höhere Fehlersicherheit zu erreichen. Im dargestellten Ausführungsbeispiel ist mit der Bezugsziffer 46a ein erster Codeteil bezeichnet, der mit dem ersten Compiler 44a aus dem ersten Programmteil 45 erzeugt wurde und der sicherheitsrelevante logische Abhängigkeiten in maschinenlesbarer Form definiert. Bezugsziffer 46b bezeichnet einen zweiten Codeteil, den der zweite Compiler 44b aus dem ersten Programmteil 45 erzeugt hat. Der zweite Codeteil 46b repräsentiert dieselben logischen Abhängigkeiten wie der erste Codeteil 46a, allerdings in einer redundanten und vorzugsweise diversitären Implementierung. In einigen bevorzugten Ausführungsbeispielen können die Codeteile 46a, 46b für unterschiedliche Prozessoren kompiliert sein. Prinzipiell ist jedoch eine Ausführung der Codeteile 46a, 46b auf einem Prozessor oder auf mehreren typgleichen Prozessoren denkbar.

[0051] Mit der Bezugsziffer 48 ist ein weiterer Codeteil bezeichnet, der in diesem Ausführungsbeispiel mit Hilfe des ersten Compilers 44a aus einem weiteren Programmteil 49 erzeugt wurde. Der weitere Codeteil 48 repräsentiert logische Abhängigkeiten zwischen ausgewählten Sensorsignalen und Aktorsignalen, die keine sicherheitsrelevanten Funktionen betreffen. Mithin ist der Codeteil 48 in diesem Ausführungsbeispiel ein Codeteil für einen Standard-Prozess, wohingegen die redundanten Codeteile 46a, 46b zum Steuern eines sicherheitsrelevanten Prozesses dienen.

[0052] Mit der Bezugsziffer 50 ist ein Binder bezeichnet, mit dessen Hilfe mehrere Codeteile 46a, 48 zu einem ausführbaren Programmcode für die Steuerung 28 zusammengefügt werden können. Es sei angemerkt, dass das neue Verfahren über die hier lediglich beispielhaft dargestellten Codeteile hinaus eine Vielzahl von Codeteilen erzeugen kann, von denen einige sicherheitsrelevant sind und andere nicht-sicherheitsrelevant sind. Typischerweise fügt der Binder 50 eine Vielzahl von Codeteilen zu einem ausführbaren Programmcode 52 zusammen. Insbesondere ist es in einigen Ausführungsbeispielen möglich, dass mehrere sicherheitsrelevante Codeteile zu einem ausführbaren Programmcode zusammengefügt werden.

[0053] Das Programmiertool 36 besitzt eine erste Schnittstelle 54, über die der ausführbare Programmcode 52 in einen ersten Speicher 56 der Steuereinheit 28 übertragen werden kann. In bevorzugten Ausführungsbeispielen ist der erste Speicher 56 ein nichtflüchtiger Speicher, beispielsweise in Form eines EEPROM, der in der Steuereinheit 28 fest verbaut ist. Alternativ oder ergänzend kann der erste nichtflüchtige Speicher 56 ein Flashspeicher sein, beispielsweise in Form einer SD-Karte.

[0054] Mit der Bezugsziffer 58 ist ein zweiter Speicher in der Steuereinheit 28 bezeichnet, der in den bevorzugten Ausführungsbeispielen ein flüchtiger Speicher ist. Das Programmiertool 36 besitzt eine weitere Schnittstelle 60, über die insbesondere ein geänderter erster Codeteil 61 a sowie ein geänderter zweiter Codeteil 61 b in den Speicher 58 übertragen werden kann.

[0055] Mit den Bezugsziffern 62a, 62b sind hier zwei Mikroprozessoren bezeichnet, die in den bevorzugten Ausführungsbeispielen der Steuereinheit 28 redundant zueinander arbeiten. Ein Doppelpfeil 64 symbolisiert, dass die Mikroprozessoren 62a, 62b ihre Arbeitsergebnisse miteinander vergleichen und/oder sich gegenseitig überprüfen können. In bevorzugten Ausführungsbeispielen führt der eine Mikroprozessor 62a den ersten sicherheitsrelevanten Codeteil 46a und den nicht-sicherheitsrelevanten Codeteil 48 aus, während der zweite Mikroprozessor 62b den zweiten sicherheitsrelevanten Codeteil 46b ausführt. Im Allgemeinen ist es bevorzugt, wenn zumindest die sicherheitsrelevanten Codeteile von der Steuereinheit 28 zweikanalig-redundant ausgeführt werden, um eine hohe Fehlersicherheit entsprechend den Anforderungen von SIL3 und/oder PL e im Sinne der eingangs genannten Normen zu erreichen. Prinzipiell ist eine fehlersichere Realisierung der Steuereinheit 28 jedoch auch in anderer Weise möglich.

[0056] Mit der Bezugsziffer 66 ist eine I/O-Schnittstelle bezeichnet, über die die Steuereinheit 28 aktuelle Prozesswerte von Sensoren 30 einlesen kann und aktuelle Prozesswerte zum Steuern der Aktoren 32 ausgeben kann. In bevorzugten Ausführungsbeispielen beinhaltet die I/O-Schnittstelle eine Busschnittstelle zum Anschluss der Steuereinheit 28 an das Kommunikationsnetzwerk 34.

[0057] In einigen Ausführungsbeispielen beinhaltet die I/O-Schnittstelle 66 eine Schnittstelle zum Zuführen einer Anwenderidentifikation 68. In einigen Ausführungsbeispielen kann die Anwenderidentifikation die Eingabe eines Passwortes und/oder das Einlesen eines elektronischen und/oder mechanischen Schlüssels beinhalten. Alternativ oder ergänzend hierzu kann die Schnittstelle zum Einlesen der Anwenderidentifikation auch in dem Programmiertool 36 implementiert sein (hier nicht dargestellt).

[0058] Bei der Bezugsziffer 70 ist eine optische Anzeige dargestellt, die beispielsweise in Form einer Signalleuchte realisiert ist. Die Anzeige 70 signalisiert in bevorzugten Ausführungsbeispielen, wenn ein geänderter erster Codeteil 61 a von der Steuereinheit 28 ausgeführt wird, wie im Folgenden erläutert ist.

[0059] Mit der Bezugsziffer 72 ist ein weiterer nichtflüchtiger Speicher bezeichnet, in dem die sogenannte Firmware der Steuereinheit 28 gespeichert ist. Die Firmware bildet ein Betriebssystem, mit dessen Hilfe die Steu-

ereinheit 28 in die Lage versetzt wird, Codeteile 46a, 46b, 48, die über das Programmiertool 36 in die Speicher 56, 58 geladen wurden, auszuführen. In bevorzugten Ausführungsbeispielen ist in der Firmware eine definierte Zeitspanne 74 hinterlegt, die die Ausführung eines geänderten ersten Codeteils 61 a und eines geänderten zweiten Codeteils 61 b zeitlich begrenzt.

[0060] Jeder der Mikroprozessoren 62a, 62b besitzt in diesem Ausführungsbeispiel einen Programmzeiger 76, der in an sich bekannter Weise auf den jeweils nächsten auszuführenden Befehlscode innerhalb des zyklisch ausgeführten Programmcodes verweist. In einigen bevorzugten Ausführungsbeispielen wird der Programmzeiger 76 in jedem der beiden Mikroprozessoren 62a, 62b mit Hilfe der Firmware 72 und mit Hilfe des Programmiertools 36 so verändert, dass der Prozessor 62a temporär einen geänderten ersten Codeteil 61 a und der Prozessor 62b temporär einen geänderten zweiten Codeteil 61 b ausführt.

[0061] Fig. 3 erläutert ein bevorzugtes Ausführungsbeispiel des neuen Verfahrens anhand eines vereinfachten Flussdiagramms. Gemäß Schritt 80 erstellt ein Anwender, der die Sicherheitssteuerung 24 programmieren möchte, zumindest einen ersten Programmteil, der sicherheitsrelevante logische Abhängigkeiten zwischen ausgewählten Sensorsignalen und ausgewählten Aktorsignalen definiert. Kurz gesagt erstellt der Anwender zumindest einen FS-Programmteil. Der FS-Programmteil kann in praktischen Ausführungsbeispielen aus einer Vielzahl von FS-Programmmodulen bestehen, so dass der erste Programmteil in weitere Unterprogrammteile unterteilt sein kann.

[0062] Gemäß Schritt 82 erstellt der Anwender zumindest einen weiteren Programmteil, in dem typischerweise keine sicherheitsrelevanten logischen Abhängigkeiten zwischen Sensorsignalen und Aktorsignalen definiert sind. Kurz gesagt erstellt der Anwender im Schritt 82 einen ST-Programmteil.

[0063] Gemäß Schritt 84 werden die Programmteile aus den Schritten 80, 82 kompiliert und zu einem ausführbaren Programmcode für die Sicherheitssteuerung 24 gebunden. In bevorzugten Ausführungsbeispielen wird der erste Programmteil (FS-Programmteil) mit Hilfe der zwei Compiler 44a, 44b zweifach kompiliert, um zwei redundante FS-Codeteile zu erzeugen. In einigen Ausführungsbeispielen wird nur einer der beiden redundanten FS-Codeteile mit dem ST-Codeteil aus Schritt 82 gebunden. Prinzipiell kann der ST-Codeteil in anderen Ausführungsbeispielen jedoch jeweils mit einem der beiden FS-Codeteile zu einem ausführbaren Programmcode gebunden werden.

[0064] Gemäß Schritt 86 wird der ausführbare Programmcode in den ersten Speicher der Sicherheitssteuerung heruntergeladen. Anschließend wird die Sicherheitssteuerung 24 gemäß Schritt 88 in Betrieb genommen, indem der heruntergeladene Programmcode mit Hilfe der Prozessoren in der Sicherheitssteuerung ausgeführt wird. In an sich bekannter Weise führt die Sicherheitssteuerung 24 den Programmcode aus dem ersten Speicher zyklisch wiederholt aus, wobei sie jeweils aktuelle Prozesswerte der Sensoren 30 einliest und in jedem Zyklus jeweils aktuelle Prozesswerte für die Aktoren 32 bestimmt und über die I/O-Schnittstelle an die Aktoren ausgibt (Schritt 89).

[0065] Entsprechend einem bevorzugten Ausführungsbeispiel kann der Anwender während der Ausführung des Programmcodes eine Änderung im ersten Programmteil vornehmen (Schritt 90), den geänderten ersten Programmteil erneut kompilieren (Schritt 92) und den so erhaltenen geänderten Codeteil gemäß Schritt 94 in den zweiten Speicher der Sicherheitssteuerung herunterladen. In den bevorzugten Ausführungsbeispielen ist zuvor eine Anwenderidentifikation erforderlich, die beispielsweise eine Passwortabfrage 96 beinhaltet. Wie bereits oben angedeutet, kann der Schritt 92 eine zweifache Kompilierung des geänderten ersten Programmteils beinhalten, um zwei redundante geänderte Codeteile 61a, 61 b zu erhalten.

[0066] Gemäß Schritt 98 wird das Herunterladen des geänderten ersten Codeteils mit einem Zeitstempel protokolliert. In bevorzugten Ausführungsbeispielen wird das Protokoll zusammen mit dem Zeitstempel in einem nichtflüchtigen Speicher des Programmiertools 36 und/oder der Sicherheitssteuerung 24 abgespeichert. Ferner wird gemäß Schritt 100 die Anzeige 70 aktiviert.

[0067] Gemäß Schritt 102, der auch vor dem Aktivieren der Anzeige erfolgen kann, wird der Programmzeiger des zumindest einen Prozessors der Sicherheitssteuerung mit Hilfe des Programmiertools 36 und/oder der Firmware 72 so geändert, dass er nunmehr auf den geänderten ersten Codeteil im flüchtigen Speicher 58 verweist, sobald im Rahmen der zyklischen Ausführung des Programmcodes aus dem ersten Speicher 56 der sicherheitsrelevante erste Codeteil ausgeführt werden soll. Infolgedessen wird anstelle des originären ersten Codeteils 46a aus dem ersten Speicher 56 nunmehr der geänderte erste Codeteil 61a aus dem flüchtigen Speicher 58 ausgeführt. In den bevorzugten Ausführungsbeispielen wird gleichermaßen der redundante geänderte zweite Codeteil 61 b aus dem flüchtigen Speicher 58 anstelle des redundanten originären zweiten Codeteils 46b aus dem Speicher 56 ausgeführt.

[0068] Die zyklische Ausführung des Programmcodes inklusive der geänderten sicherheitsrelevanten Codeteile ist in den bevorzugten Ausführungsbeispielen zeitlich begrenzt. Dementsprechend wird gemäß Schritt 104 abgefragt, ob die Sicherheitssteuerung durch einen Softwarereset oder einen Neustart zu dem originären Programmcode zurückkehren soll. Ist dies nicht der Fall, wird gemäß Schritt 106 überprüft, ob das maximale Zeitlimit 74 für die temporäre Ausführung des geänderten Programmcodes erreicht ist. Ist dies der Fall, sorgt die Firmware der Sicherheitssteuerung 24 in den bevorzugten Ausführungsbeispielen dafür, dass die Sicherheitssteuerung und mithin die Anlage 10 einen sicheren Zustand einnimmt (Schritt 108), aus dem heraus ein manueller

Neustart erforderlich ist. Entsprechend Schritte 110 führt die Sicherheitssteuerung nach dem manuellen Neustart wieder den originären, nicht-geänderten Programmcode aus.

[0069] Wenn der Anwender den geänderten Programmcode dauerhaft ausgeführt haben möchte, muss er gemäß Schritt 112 den geänderten ersten Programmteil zusammen mit dem weiteren Programmteil erneut kompilieren und binden. Der geänderte Programmcode wird somit als neuer originärer Programmcode in den nicht-flüchtigen Speicher 56 der Sicherheitssteuerung 24 übertragen.

[0070] In den bevorzugten Ausführungsbeispielen können mit dem neuen Verfahren lediglich logische Abhängigkeiten in dem ersten Programmteil 45 geändert werden, nicht jedoch I/O-Zuordnungen zwischen physischen Anschlüssen der Sicherheitssteuerung 24 und Prozessvariablen (sogenanntes I/O-Mapping). Auch eine Änderung in der Hardware der Sicherheitssteuerung 24 ist in den bevorzugten Ausführungsbeispielen ausgeschlossen, solange die Sicherheitssteuerung 24 den Programmcode ausführt. Mit anderen Worten müssen solche Änderungen vom Anwender durchgeführt werden, wenn die Sicherheitssteuerung 24 außer Betrieb gesetzt ist und keine aktuellen Steuerfunktionen ausführt.

[0071] In allen bevorzugten Ausführungsbeispielen sind die Änderung im ersten Programmteil und die daraus folgende Änderung des ersten Codeteils nicht-persistent. Dies bedeutet, dass der ursprünglich auf die Sicherheitssteuerung übertragene Programmcode nicht überschrieben wird, sondern unverändert beibehalten bleibt. In allen bevorzugten Ausführungsbeispielen werden die Änderungen nach dem neuen Verfahren nur im flüchtigen Speicher 58 durchgeführt. Infolgedessen kehrt die Sicherheitssteuerung nach Ausführen eines Resetbefehls oder durch einen Neustart, beispielsweise nach Ausschalten der Versorgungsspannung, zu dem originären Programmcode zurück. Eine dauerhafte Änderung des Programmcodes erfordert in den bevorzugten Ausführungsbeispielen einen erneuten Download des vollständigen kompilierten und gebundenen Programmcodes.

[0072] Wie bereits weiter oben angedeutet wurde, ermittelt das Programmiertool 36 in den bevorzugten Ausführungsbeispielen sämtliche Steuereinheiten 28, 28', 28" etc., auf denen ein geänderter erster Codeteil 61a ausgeführt werden soll. Anschließend stellt das Programmiertool 36 in den bevorzugten Ausführungsbeispielen den geänderten ersten Codeteil (und vorzugsweise auch den geänderten zweiten Codeteil für die redundante Ausführung) zur Übernahme in alle betroffenen Steuereinheiten zur Verfügung. Konsequenterweise werden anschließend die Programmzeiger in allen betroffenen Steuereinheiten auf den geänderten Codeteil temporär umgeleitet.

[0073] In organisatorischer Hinsicht liegt es in der Verantwortung des Anwenders, eine spezielle Risikoanalyse für die Änderung im ersten Programmteil 45 durchzuführen. Gegebenenfalls muss der Anwender dafür Sorge tragen, dass bei einem verminderten Sicherheitslevel Gefahren für Personen im Umfeld der gesteuerten Anlage auf andere Weise beherrscht werden.

Patentansprüche

1. Verfahren zum Programmieren einer Sicherheitssteuerung (24), die eine Vielzahl von Eingängen zum Aufnehmen von Sensorsignalen (30), eine Vielzahl von Ausgängen zum Ausgeben von Aktorsignalen (32) und zumindest einen Prozessor (62a, 62b) zum Ausführen von Programmcode aufweist, mit den Schritten:

   - Erstellen eines Anwenderprogramms, das logische Abhängigkeiten zwischen den Sensorsignalen (30) und den Aktorsignalen (32) definiert, unter Verwendung eines Programmeditors (36), wobei das Anwenderprogramm einen ersten Programmteil (45) und zumindest einen weiteren Programmteil (49) aufweist, und wobei der erste Programmteil (45) sicherheitsrelevante logische Abhängigkeiten zwischen ausgewählten Sensorsignalen und Aktorsignalen definiert,
   - Kompilieren und Binden des ersten und des zumindest einen weiteren Programmteils, um einen ausführbaren originären Programmcode für den zumindest einen Prozessor (62a, 62b) zu erhalten, wobei der originäre Programmcode einen originären ersten Codeteil (46a) aufweist, der den ersten Programmteil repräsentiert, und zumindest einen originären weiteren Codeteil (48), der den zumindest einen weiteren Programmteil repräsentiert,
   - Übertragen des originären Programmcodes in einen ersten Speicher (56) der Sicherheitssteuerung (24),
   - Ausführen des originären Programmcodes mit Hilfe des zumindest einen Prozessors (62a, 62b), um die Aktorsignale in Abhängigkeit von den Sensorsignalen zu erzeugen,
   - Ändern des ersten Programmteils (45) unter Verwendung des Programmeditors (36), um einen geänderten ersten Programmteil zu erhalten, und
   - Kompilieren des geänderten ersten Programmteils, um einen geänderten ersten Codeteil (61a) zu erhalten,

   dadurch gekennzeichnet, dass der geänderte ersten Codeteil (61a) in einen zweiten Speicher (58) der Sicherheitssteuerung (24) übertragen wird, während der originäre Programmcode mit Hilfe des zumindest einen Prozessors (62a, 62b) ausgeführt wird, und dass der geänderte ersten Codeteil (61a) anschlie-

ßend in Ergänzung zu dem originären weiteren Codeteil (48) und anstelle des originären ersten Codeteils (46a) mit Hilfe des zumindest einen Prozessors (62a, 62b) ausgeführt wird, um die Aktorsignale in Abhängigkeit von dem geänderten ersten Programmteil zu erzeugen.

2. Verfahren nach Anspruch 1, **dadurch gekennzeichnet, dass** der geänderte erste Codeteil (61a) nur während einer begrenzten Zeitspanne (74) anstelle des originären ersten Codeteils (46a) ausgeführt wird.

3. Verfahren nach Anspruch 2, **dadurch gekennzeichnet, dass** die Sicherheitssteuerung (24) nach Ablauf der begrenzten Zeitspanne (74) einen definierten sicheren Zustand einnimmt.

4. Verfahren nach einem der Ansprüche 1 bis 3, **dadurch gekennzeichnet, dass** der erste Speicher (56) ein nichtflüchtiger Speicher ist, und dass der zweite Speicher (58) ein flüchtiger Speicher ist.

5. Verfahren nach einem der Ansprüche 1 bis 4, **dadurch gekennzeichnet, dass** ein Programmzeiger (76) der Sicherheitssteuerung (24) geändert wird, um den geänderten ersten Codeteil (61 a) anstelle des originären ersten Codeteils (46a) auszuführen.

6. Verfahren nach einem der Ansprüche 1 bis 5, **dadurch gekennzeichnet, dass** der zumindest eine Prozessor (62a, 62b) den originären Programmcode zyklisch ausführt, und dass der geänderte ersten Codeteil (61 a) nach Abschluss eines Programmzyklus anstelle des originären ersten Codeteils (46a) ausgeführt wird.

7. Verfahren nach einem der Ansprüche 1 bis 6, **dadurch gekennzeichnet, dass** die Sicherheitssteuerung (24) eine optische Anzeige (70) aufweist, mit der die Ausführung des geänderten ersten Codeteils (61 a) signalisiert wird.

8. Verfahren nach einem der Ansprüche 1 bis 7, **dadurch gekennzeichnet, dass** der geänderte erste Programmteil (61 a) zusammen mit dem zumindest einen weiteren Programmteil (48) kompiliert und gebunden wird, um einen geänderten ausführbaren Programmcode zu erhalten, und dass der geänderte ausführbare Programmcode in den ersten Speicher (56) übertragen wird, nachdem die Ausführung des originären Programmcodes gestoppt wurde.

9. Verfahren nach einem der Ansprüche 1 bis 8, **dadurch gekennzeichnet, dass** der geänderte erste Programmteil zweifach kompiliert wird, um einen geänderten ersten Codeteil (61 a) und einen geänderten zweiten Codeteil (61 b) zu erzeugen, wobei der geänderte erste Codeteil und der geänderte zweite Codeteil in den zweiten Speicher (58) übertragen werden und mit Hilfe des zumindest einen Prozessors (62a, 62b) redundant ausgeführt werden.

10. Verfahren nach einem der Ansprüche 1 bis 9, **dadurch gekennzeichnet, dass** die Sicherheitssteuerung (24) eine verteilte Sicherheitssteuerung mit einer Vielzahl von vernetzten Steuereinheiten (28, 28', 28", 28"') ist, auf denen jeweils ein individueller Programmcode ausgeführt wird, wobei der geänderte erste Codeteil an zumindest zwei verschiedene Steuereinheiten übertragen wird, während der originäre Programmcode ausgeführt wird.

11. Verfahren nach einem der Ansprüche 1 bis 10, **dadurch gekennzeichnet, dass** die Änderung des ersten Programmteils (45) in Abhängigkeit von einer Anwenderidentifizierung (68) freigegeben oder gesperrt wird.

12. Verfahren nach einem der Ansprüche 1 bis 11, **dadurch gekennzeichnet, dass** die Übertragung des geänderten ersten Codeteils (61 a) an den zweiten Speicher (58) in einem weiteren Speicher (72) zusammen mit einem Datumsstempel protokolliert wird.

13. Sicherheitssteuerung zum fehlersicheren Steuern eines sicherheitskritischen Prozesses, mit einer Vielzahl von Eingängen zum Aufnehmen von Sensorsignalen, mit einer Vielzahl von Ausgängen zum Ausgeben von Aktorsignalen, mit zumindest einem Prozessor (62a, 62b) zum zyklischen Ausführen von Programmcode, mit einem ersten nichtflüchtigen Speicher (56), mit einem zweiten flüchtigen Speicher (58), und mit einem Programmiertool (36) zum Erstellen eines Anwenderprogramms, das logische Abhängigkeiten zwischen ausgewählten Sensorsignalen und ausgewählten Aktorsignalen definiert,
wobei das Programmiertool (36) einen Programmeditor (42) aufweist, der dazu ausgebildet ist, einen ersten Programmteil und zumindest einen weiteren Programmteil zu erzeugen, wobei der erste Programmteil sicherheitsrelevante logische Abhängigkeiten zwischen einigen ausgewählten Sensorsignalen und Aktorsignalen definiert,
wobei das Programmiertool (36) ferner einen Compiler (44) aufweist, der dazu ausgebildet ist, den ersten und den zumindest einen weiteren Programmteil zu kompilieren und zu einem ausführbaren originären Programmcode für den zumindest einen Prozessor (62a, 62b) binden, wobei der originäre Programmcode einen originären ersten Codeteil (46a) aufweist, der den ersten Programmteil repräsentiert, und zumindest einen originären weiteren Codeteil (48), der den zumindest einen weiteren Programmteil repräsentiert, und

wobei das Programmiertool (36) eine Schnittstelle (54) zum Übertragen des originären Programmcodes in den ersten Speicher aufweist,
dadurch gekennzeichnet, dass das Programmiertool (36) dazu ausgebildet ist, den ersten Programmteil separat von dem weiteren Programmteil zu ändern und zu kompilieren, um einen geänderten ersten Codeteil zu erhalten, und dass das Programmiertool (36) eine weitere Schnittstelle (60) aufweist, die dazu ausgebildet ist, den geänderten ersten Codeteil in den zweiten Speicher (58) zu übertragen, während der originäre Programmcode mit Hilfe des zumindest einen Prozessors (62a, 62b) ausgeführt wird, wobei die Sicherheitssteuerung (24) dazu ausgebildet ist, den geänderten ersten Codeteil in Ergänzung zu dem originären weiteren Codeteil und anstelle des originären ersten Codeteils mit Hilfe des zumindest einen Prozessors (62a, 62b) auszuführen, um die Aktorsignale in Abhängigkeit von dem geänderten ersten Programmteil zu erzeugen.

14. Computerprogramm für eine Sicherheitssteuerung zum fehlersicheren Steuern eines sicherheitskritischen Prozesses, wobei die Sicherheitssteuerung eine Vielzahl von Eingängen zum Aufnehmen von Sensorsignalen, eine Vielzahl von Ausgängen zum Ausgeben von Aktorsignalen, zumindest einen Prozessor zum zyklischen Ausführen von Programmcode, einen ersten nichtflüchtigen Speicher, einen zweiten flüchtigen Speicher, und ein Programmiertool zum Erstellen eines Anwenderprogramms aufweist, das logische Abhängigkeiten zwischen ausgewählten Sensorsignalen und ausgewählten Aktorsignalen definiert, wobei das Computerprogramm dazu ausgebildet ist, ein Verfahren nach einem der Ansprüche 1 bis 12 auszuführen, wenn das Computerprogramm auf der Sicherheitssteuerung implementiert ist.

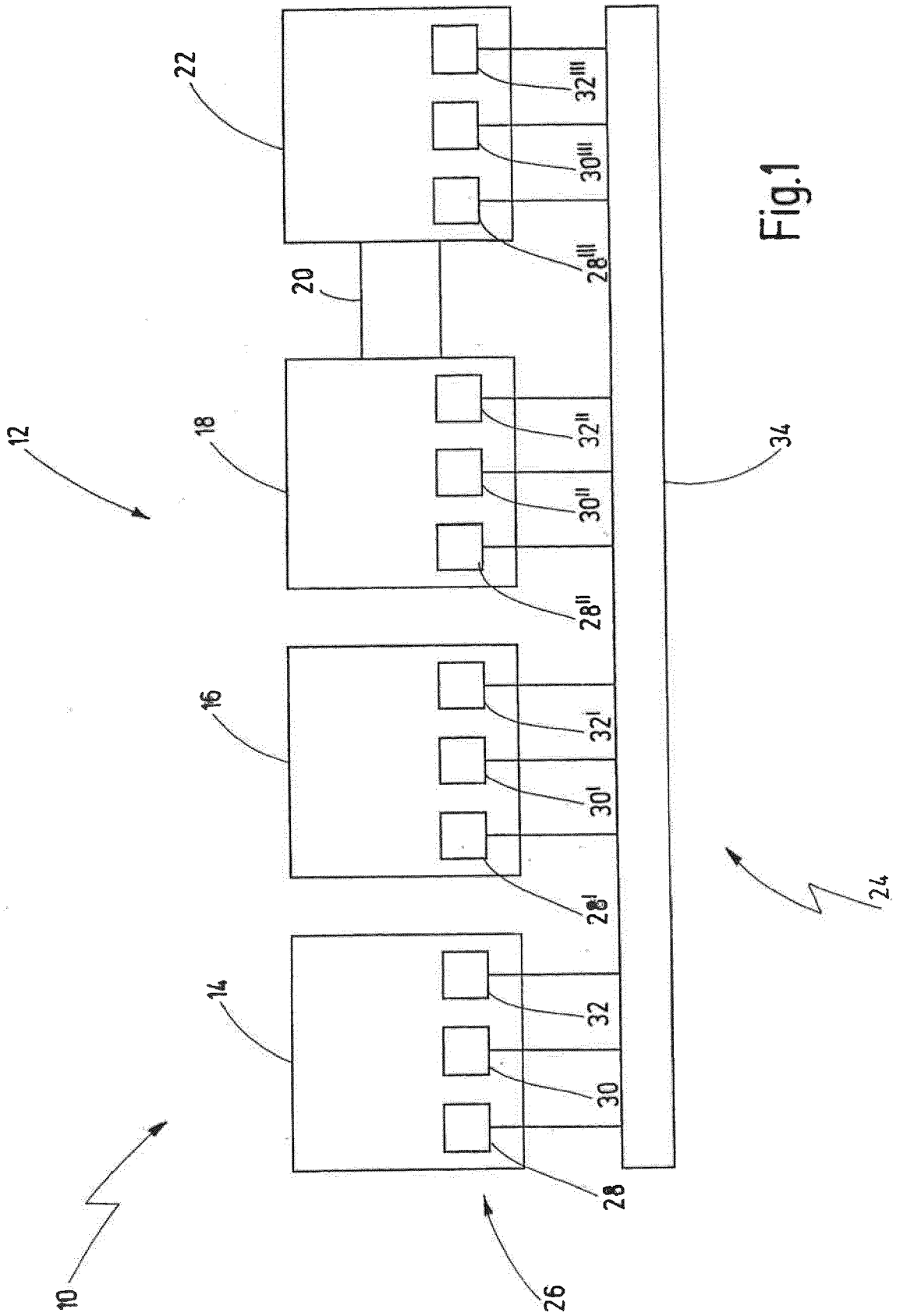

Fig.1

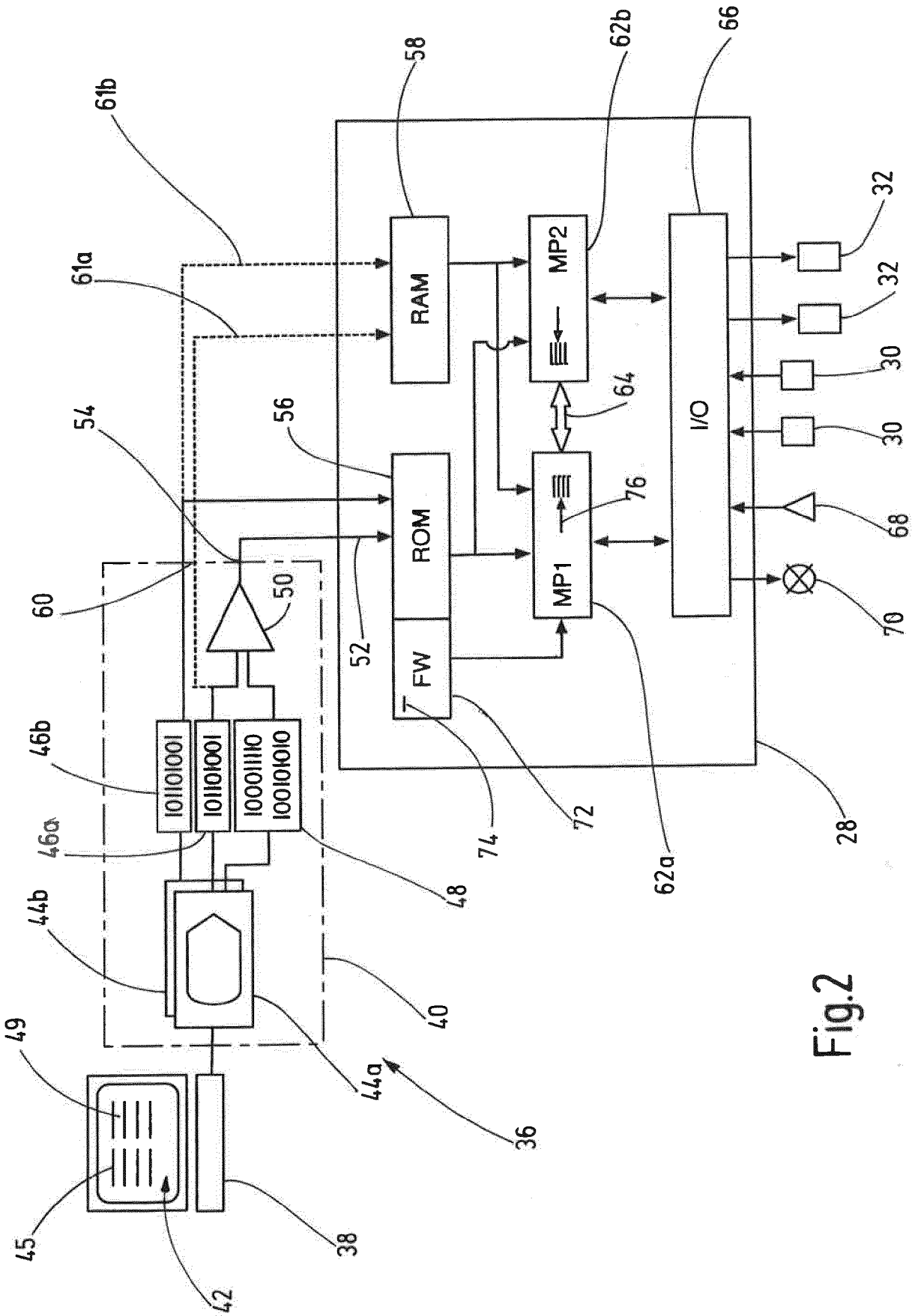
61b
61a
58
62b
66
RAM
MP2
I/O
32
32
30
30
64
54
56
ROM
MP1
76
68
60
50
52
FW
70
46b
46a
48
74
72
62a
28
44b
40
49
44a
36
45
38
42

Fig.2

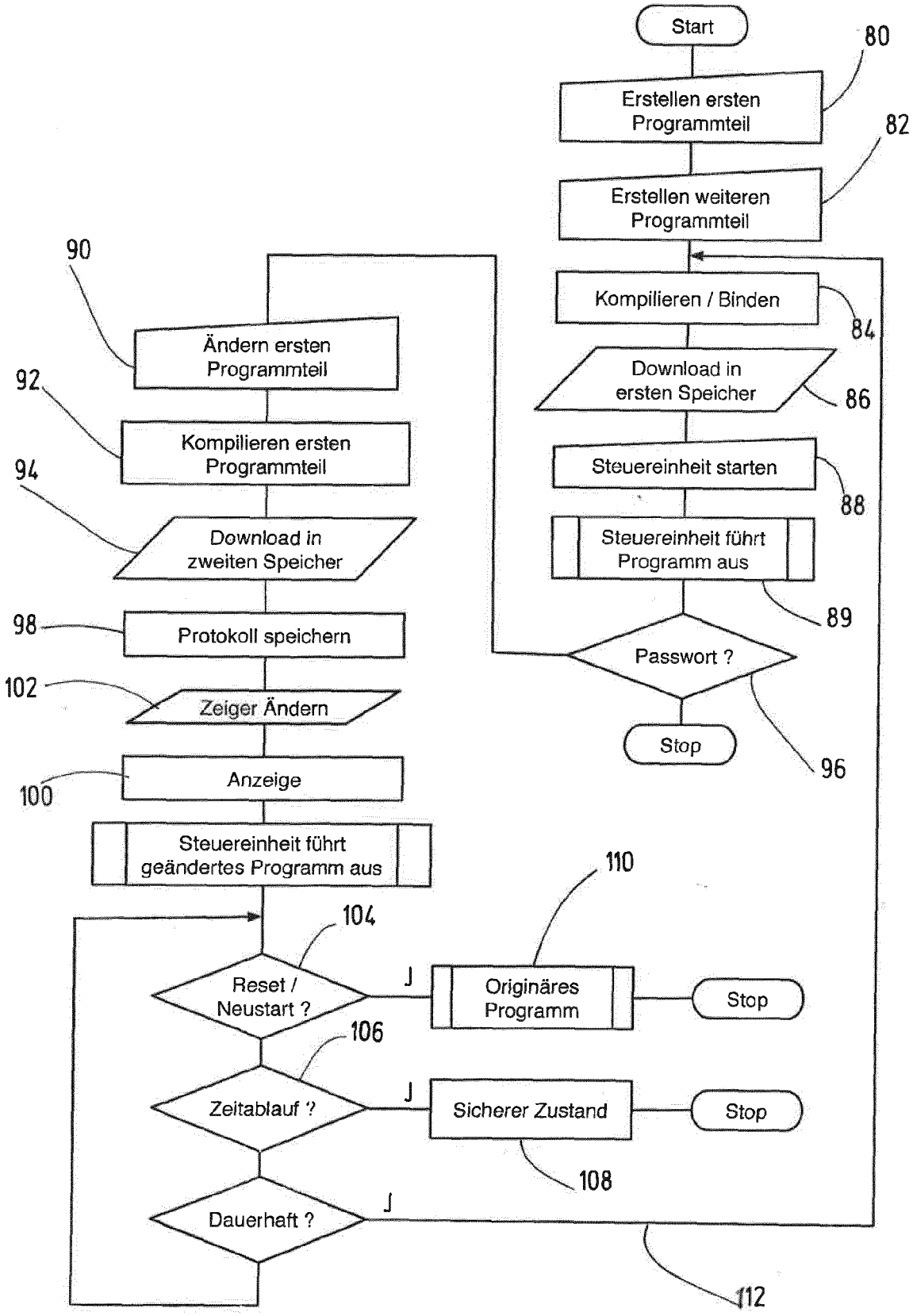
Start
80
Erstellen ersten Programmteil
82
Erstellen weiteren Programmteil
Kompilieren / Binden
84
Download in ersten Speicher
86
Steuereinheit starten
88
Steuereinheit führt Programm aus
89
Passwort ?
96
Stop
90
Ändern ersten Programmteil
92
Kompilieren ersten Programmteil
94
Download in zweiten Speicher
98
Protokoll speichern
102
Zeiger Ändern
100
Anzeige
Steuereinheit führt geändertes Programm aus
110
104
Reset / Neustart ?
J
Originäres Programm
Stop
106
Zeitablauf ?
J
Sicherer Zustand
Stop
108
Dauerhaft ?
J
112

Fig.3

Europäisches Patentamt
European Patent Office
Office européen des brevets

EUROPÄISCHER RECHERCHENBERICHT

Nummer der Anmeldung
EP 16 19 7755

EINSCHLÄGIGE DOKUMENTE

Kategorie	Kennzeichnung des Dokuments mit Angabe, soweit erforderlich, der maßgeblichen Teile	Betrifft Anspruch	KLASSIFIKATION DER ANMELDUNG (IPC)
X Y	US 2012/078392 A1 (WOEHRLE STEFAN [DE] ET AL) 29. März 2012 (2012-03-29) * Absätze [0002], [0007], [0008], [0011], [0019] - [0021], [0049] - [0052], [0079] - [0116] * * Abbildungen 1-4 * -----	1-14 1-14	INV. G05B19/042
X	DE 10 2008 060003 A1 (PILZ GMBH & CO KG [DE]) 27. Mai 2010 (2010-05-27) * Absätze [0001] - [0012], [0031], [0040] - [0057], [0066], [0126] - [0143], [0151] - [0153], [0176], [0184] * * Abbildungen 2,3 * -----	1-14	
X	DE 10 2009 011679 A1 (PILZ GMBH & CO KG [DE]) 26. August 2010 (2010-08-26) * Absätze [0001] - [0006], [0037] - [0058], [0081] - [0093] * -----	1-13	
X	EP 1 515 205 A2 (ROCKWELL AUTOMATION TECH INC [US]) 16. März 2005 (2005-03-16) * Absätze [0001] - [0021], [0038] - [0065] * * Abbildung 1 * -----	1-14	RECHERCHIERTE SACHGEBIETE (IPC) G05B G06F
Y	US 5 970 243 A (KLEIN MICHAEL T [US] ET AL) 19. Oktober 1999 (1999-10-19) * das ganze Dokument * -----	1-14	
A	US 2010/192122 A1 (ESFAHAN HAMID M [US] ET AL) 29. Juli 2010 (2010-07-29) * Absätze [0006], [0007], [0036], [0045] - [0054], [0082], [0083] * -----	1,13,14	

Der vorliegende Recherchenbericht wurde für alle Patentansprüche erstellt

Recherchenort	Abschlußdatum der Recherche	Prüfer
München	28. März 2017	Postemer, Patricia

KATEGORIE DER GENANNTEN DOKUMENTE

X : von besonderer Bedeutung allein betrachtet
Y : von besonderer Bedeutung in Verbindung mit einer anderen Veröffentlichung derselben Kategorie
A : technologischer Hintergrund
O : nichtschriftliche Offenbarung
P : Zwischenliteratur

T : der Erfindung zugrunde liegende Theorien oder Grundsätze
E : älteres Patentdokument, das jedoch erst am oder nach dem Anmeldedatum veröffentlicht worden ist
D : in der Anmeldung angeführtes Dokument
L : aus anderen Gründen angeführtes Dokument

& : Mitglied der gleichen Patentfamilie, übereinstimmendes Dokument

EPO FORM 1503 03.82 (P04C03)

ANHANG ZUM EUROPÄISCHEN RECHERCHENBERICHT ÜBER DIE EUROPÄISCHE PATENTANMELDUNG NR. EP 16 19 7755

In diesem Anhang sind die Mitglieder der Patentfamilien der im obengenannten europäischen Recherchenbericht angeführten Patentdokumente angegeben.
Die Angaben über die Familienmitglieder entsprechen dem Stand der Datei des Europäischen Patentamts am
Diese Angaben dienen nur zur Unterrichtung und erfolgen ohne Gewähr.

28-03-2017

Im Recherchenbericht angeführtes Patentdokument	Datum der Veröffentlichung	Mitglied(er) der Patentfamilie	Datum der Veröffentlichung
US 2012078392 A1	29-03-2012	CN 102549508 A	04-07-2012
		DE 102009019088 A1	11-11-2010
		EP 2422243 A1	29-02-2012
		HK 1166525 A1	08-11-2013
		JP 2012524354 A	11-10-2012
		US 2012078392 A1	29-03-2012
		WO 2010121797 A1	28-10-2010
DE 102008060003 A1	27-05-2010	CN 102292680 A	21-12-2011
		DE 102008060003 A1	27-05-2010
		EP 2353051 A1	10-08-2011
		JP 2012510099 A	26-04-2012
		US 2012004744 A1	05-01-2012
		WO 2010060575 A1	03-06-2010
DE 102009011679 A1	26-08-2010	CN 102414627 A	11-04-2012
		DE 102009011679 A1	26-08-2010
		EP 2399174 A1	28-12-2011
		ES 2493068 T3	11-09-2014
		JP 5636378 B2	03-12-2014
		JP 2012518823 A	16-08-2012
		US 2012036493 A1	09-02-2012
		WO 2010094466 A1	26-08-2010
EP 1515205 A2	16-03-2005	EP 1515205 A2	16-03-2005
		US 2005060508 A1	17-03-2005
US 5970243 A	19-10-1999	KEINE	
US 2010192122 A1	29-07-2010	CN 101788820 A	28-07-2010
		DE 102010005648 A1	28-10-2010
		US 2010192122 A1	29-07-2010

EPO FORM P0461

Für nähere Einzelheiten zu diesem Anhang : siehe Amtsblatt des Europäischen Patentamts, Nr.12/82

IN DER BESCHREIBUNG AUFGEFÜHRTE DOKUMENTE

Diese Liste der vom Anmelder aufgeführten Dokumente wurde ausschließlich zur Information des Lesers aufgenommen und ist nicht Bestandteil des europäischen Patentdokumentes. Sie wurde mit größter Sorgfalt zusammengestellt; das EPA übernimmt jedoch keinerlei Haftung für etwaige Fehler oder Auslassungen.

In der Beschreibung aufgeführte Patentdokumente

- DE 102009019096 A1 **[0003]**